ARMY RESEARCH LABORATORY

A Proposal for a Taxonomy for Vulnerabilities in Supervisory Control and Data Acquisition (SCADA) Systems

by Sidney C Smith

ARL-TR-7091 September 2014

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Aberdeen Proving Ground, MD 21005-5067

ARL-TR-7091 **September 2014**

A Proposal for a Taxonomy for Vulnerabilities in Supervisory Control and Data Acquisition (SCADA) Systems

Sidney C Smith

Computational and Informational Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
September 2014	Final	August 2013–June 2014

4. TITLE AND SUBTITLE
A Proposal for a Taxonomy for Vulnerabilities in Supervisory Control and Data Acquisition (SCADA) Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Sidney C Smith

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-CIN-S
Aberdeen Proving Ground, MD 21005-5069

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-7091

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Author email: <sidney.c.smith24.civ@mail.mil>

14. ABSTRACT
To be useful a taxonomy must be unique, complete, and relational. Unique implies that each item should have one and only one place within the taxonomy. Complete means that everything should have a place in the taxonomy or that the taxonomy may be easily expanded to create a place for it. Relational means that similar items are grouped together in such a way as to allow us to make useful generalizations. This report will analyze existing taxonomies for uniqueness and completeness, and then propose a taxonomy for Supervisory Control and Data Acquisition and Industrial Control Systems that is unique, complete, and relational.

15. SUBJECT TERMS
SCADA, vulnerability, taxonomy, ICS, PLC

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Sidney C Smith
Unclassified	Unclassified	Unclassified	UU	32	19b. TELEPHONE NUMBER (*Include area code*) 410-278-6235

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Tables

1. Introduction

Taxonomies exist to provide a shared framework and lexicon making it possible for one to distinguish between *Malus domestica* and *Citrus sinensis* because without a clear shared taxonomy one could easily find oneself comparing apples to oranges. I used the binomial name for apples and orange to illustrate several key points in one of the best known taxonomies. In 1735 Carolus Linnaeus published Systema Naturae creating a taxonomy which serves as the basis of the classification still in use today.[1] This system is hierarchical with kingdoms, phyla, classes, orders, family, genus, and species. One of the key strengths of the Linnaeus' system is the use of trivial names or binomial names. Thus, we may refer to *Malus domestica* and not *Platae Angiosperms Eudicots Rosids Rosales Rosaceae Malus domestica*. Although the path from the root to the leaf is highly valuable, it is clearly unworkable in most applications.

To be useful a taxonomy must be unique, complete, and relational. Unique implies that each item should have one and only one place within the taxonomy. Complete means that everything should have a place in the taxonomy or that the taxonomy may be easily expanded to create a place for it. Relational means that similar items are grouped together in such a way as to allow us to make useful generalizations. In this way there is a place for everything, everything is in its place, and its location tells us something useful about it.

As we consider the use of a taxonomy for the vulnerabilities of Supervisory Control and Data Acquisition (SCADA) systems, there are issues such as attacks, controls and principles that we would like to easily link to vulnerabilities. "A 'vulnerable state' is 'any state which enables a user to read information without authorization, modify information without authorization, or grant or deny an entity access to a resource without authorization.' "[2] Attacks exploit one or more vulnerabilities; however, attacks are not vulnerabilities themselves. Controls are the systems put in place to ensure that confidentiality, integrity, availability, and authenticity of the information. A vulnerability could be defined as the absence or failure of a control. Thus, we see that there is a one-to-many relationship between attacks and vulnerabilities, but there may be a one-to-one relationship between vulnerabilities and controls. Principles are axiomatic security practices that are universally accepted. These include "least privilege", which means that actors should posses only the privileges that are necessary to complete the task and "defense in depth" which means that controls should not have a single point of failure. Vulnerabilities violate one or more of these principles. Vulnerabilities, attacks, controls, and principle all play an important role in

implementing a secure system; therefore, we would like the taxonomy to be able to assist in relating them.

The purpose that a taxonomy is intended to serve also plays a significant role in the construction of the taxonomy. The primary purpose of this taxonomy is to provide a framework for an assessment of the countermeasures currently available to protect SCADA and other Industrial Control Systems (ICS). A secondary purpose is to provide a framework for security assessments and risk scoring of SCADA and ICS. This is a highly general purpose and significantly different from the highly specialized purpose of assessing vulnerabilities in SCADA network protocols that Igure addressed.[3]

2. Background

In 1996 Bishop and Bailey published a critical analysis of vulnerability taxonomies. They used the race conditions in xterm and mkdir along with the buffer overflow in fingerd to illustrate the problems that they observed with uniqueness in the Program Analysis study, the Research into Secure Operating Systems study and Aslam's Taxonomy.[4]

In 1995 Bishop described a taxonomy of UNIX vulnerabilities. His system was composed of 6 axes: vulnerability class, time of introduction, exploitation domain, effect domain, minimum number, and source.[2] He demonstrates its uniqueness by analyzing and classifying 11 different vulnerabilities. The system is not complete in that there are vulnerabilities that will not fit into it. For example, on 2013 April 16, an attack was conducted against the Metcalf substation where telephone cables in an underground vault were cut, and snipers were able to damage several critical pieces of equipment. The substation was down for almost a month while technicians repaired the damage.[5] There is no place for physical vulnerabilities in Bishop's model. He defines 6 axes specifically wanting his taxonomy to relate to intrusion detection; however, he only uses the vulnerability class axis for this purpose. The other axes seem to be somewhat superfluous.

In 2008 Igure et al. published a survey of attack and vulnerability taxonomies specifically looking for a taxonomy that would be helpful in security assessments. They found none but identified features useful in such a taxonomy.[6]

The Common Vulnerabilities and Exposures list[7] provides a unique id for every known vulnerability; however, this taxonomy only tells us the year that a vulnerability was discovered. It provides a fine key into the database but does not help classify the problem in any meaningful way.

In 2011 Zhu et al. published a taxonomy of cyber attacks on SCADA systems. They did a good job illustrating the key differences between SCADA systems and typical information technology (IT) systems; however, they didn't actually propose a taxonomy.[8]

Tsipenyuk et al. created a taxonomy designed to aid developers in recognizing common coding errors, which they named the "Seven Pernicious Kingdoms".[9] Each "kingdom" is further subdivided into "phyla".

Igure makes a very important observation,"Several different taxonomies exist because each is mostly applicable only to a particular field of interest."[6] Going back to our Systema Naturae analogy, it is almost as if the efforts in taxonomy so far have looked at classifying all *Equidae* and *Canidae*. To define a complete taxonomy, we will need to take a few steps back to get the big picture.

3. Controls

If we accept that a vulnerability is either the absence or failure of a control, then it makes sense for us to look at control taxonomies. There exist several control taxonomies used in the assessment space. The National Security Agency (NSA) developed the Information Security (INFOSEC) Assessment Methodology (IAM) to encapsulate the lessons that they had learned over years of doing information security assessments.[10] The National Institute of Standards and Technology (NIST) published Special Publication 800-53, which contains a taxonomy of security controls used to support certification and accreditation.[11] A combination of 1 or 2 or more of these may provide a framework for a overarching taxonomy of vulnerabilities.

3.1 Sample Vulnerabilities

To evaluate the various existing taxonomies and the proposed taxonomy, we will consider the following vulnerabilities: the mkdir, xterm and fingerd vulnerability examined by Matt Bishop, along with the cable and sniper vulnerabilities exposed by the Metcalf incident.

mkdir The mkdir flaw was identified as a race condition where an operation that must be atomic is not. The orginal mkdir command first used mknode to create the directory then set the permissions. This 2-step process allowed someone to swap the node created by mknode with another file. The fix was to create the system call mkdir, which atomically does the work.[4]

xterm The xterm flaw was identified as a race condition where determining the ability to access a file and opening the file, which needs to be atomic to be secure, was done in 2 steps, and allows an external process to swap files between the determination and the open. Xterm ran setuid root to allow it to create an entry in utmp and to set the permission on the psuedo terminal. Both of these operations happen very early in the operation of xterm, and honoring least privilege would have xterm relinquish those elevated privileges as soon as they were no longer necessary. Modern versions of xterm are no longer setuid root because they have isolated the privileged operations into a separate executable that is setuid root.[4]

fingerd The Morris worm exploited a buffer overflow in the fingerd program to gain unauthorized access to computer systems on the Internet. Fingerd expected a name of not more than 512 characters; however, it did not check that the data it read from the network contained only 512 characters. Supplying more than 512 characters allowed the attacker to

overflow the buffer and write onto the stack. The access to the stack was exploited to implant machine code that gave the attacker shell access and set the return address of the current function to the implanted machine code.[4]

cable During the Metcalf incident, attackers were able to gain entrance to an underground vault near the highway. Once they had gained access to the vault, they were able to sever telephone cables in a way that would be difficult to repair.[5] This is clearly a vulnerability in the physical security of the site allowing unauthorized access to critical infrastructure.

sniper During the Metcalf incident, snipers were able to damage several pieces of critical equipment bringing the substation offline for almost a month. One might consider this a physical security vulnerability; however, requiring that organizations maintain control of an area large enough to prevent or harden each component against this kind of attack would be prohibitively expensive. It would be better to consider this as a recovery vulnerability. Some eventualities cannot be prevented and contingency plans must be in place to mitigate them.

3.2 NSA IAM

The NSA IAM divides controls into 3 classes and 18 categories as shown in Table 1.[10] These classes and categories do a great job of instructing an assessor where to look for vulnerabilities and encompass many of the vulnerabilities that were unaddressed in other taxonomies. There are still holes; in fact, none of the vulnerabilities that Bishop discussed have a clear place in this taxonomy until a patch or new revision corrected them and the Technical/Maintenance control should have corrected them, or a signature was developed and the Techincal/Malicious code protection should correct this. The closest is Technical/System Assurance, but that really speaks to certification and accreditation. The cable and sniper vulnerabilities exploited in the Metcalf incident would have been covered in “Operational/Physical Enviroment” and “Management/Contingency Planning”, respectively.

3.3 DOD Instruction 8500.2 Information Assurance

In 2003 the Department of Defense published instruction 8500.2 Information Assurance (IA) Implementation.[12] This instruction contained a catalog of 157 controls as seen in Table 2. The controls are over 10 years old and are out-of-date; however, it is the taxonomy that is of interest.

They divided the space into 8 subject areas. Each subject area had a name and a 2-letter abbreviation. The subject area contained controls. Each control had a name and a 2-letter abbreviation. Controls also had levels which were expressed as the numbers 1, 2, or 3. This gave

Table 1 NSA IAM classes and categories

Management	Technical	Operational
• INFOSEC Documentation • INFOSEC Roles and Responsibilities • Contingency Planning • Configuration Management	• Indentification and Authentication • Account Management • Session Controls • Auditing • Malicious Code Protection • Maintenance • System Assurance • Networking/Connectivity • Communication Security	• Media Controls • Labeling • Physical Environment • Personnel Security • Eduction Training and Awareness

Table 2 DODI subject areas

Abbreviation	Subject Area Name	Controls
DC	Security Design & Configuration	31
IA	Identification and Authentication	9
EC	Enclave and Computing Environment	48
EB	Enclave Boundary Defense	8
PE	Physical and Environmental	27
PR	Personnel	7
CO	Continuity	24
VI	Vulnerability and Incident Management	3

each control an abbreviation where the first 2 letters were for the subject area, the second 2 letters were the control, and the number designated the level. For example, the abbreviation ECCT-2 specificies the Enclave and Computing Environment, Encryption for Confidentiality (Data in Transit), level 2. Each control was also mapped to an IA service: Integrity, Confidentiality, or Availability.

The vulnerabilities like the ones in mkdir, xterm, and fingerd do not have a well defined home in this taxonomy. The closest we find is Security Design and Configuration/Software Quality; however, that serves as a broad category for everything contained in that taxonomy, and specifically called for validation methods to be in place to prevent them.

The vulnerabilities exposed in the Metcalf incident are covered in the Physical and Environmental and Continuity subject areas. Several confidentiality controls in the Physical and Environmental subject area are designed to ensure that only authorized personnel have access to sensitive areas and would have addressed the cable vulnerability. The continuity controls for availability seem to center around the use of an alternate site to address the sniper vulnerability. In this respect the power company performed well because power was rerouted from different sites to avoid a blackout.[5]

3.4 NIST Special Publication 800-53

In this publication NIST defines 3 classes and 18 families as shown in Table 3. The NSA IAM and NIST Security Controls share the same classes, and many of their categories and family map one to one. The NIST control taxonomy takes it one step further down by enumerating a number of controls in each of the families. The NIST taxonomy also has the advantage of containing the baseline controls for Certification and Accreditation of all federal government systems.

Table 3 NIST classes and families

Identifier	Family	Class	Controls
AC	Access Control	Technical	22
AT	Awareness and Training	Operational	5
AU	Audit and Accountability	Technical	14
CA	Security Assessment and Authorization	Management	7
CM	Configuration Management	Operational	9
CP	Contingency Planning	Operational	10
IA	Identification and Authentication	Technical	8
IR	Incident Response	Operational	8
MA	Maintenance	Operational	6
MP	Media Protection	Operational	6
PE	Physical Environment Protection	Operational	19
PL	Planning	Management	6
PS	Personnel Security	Management	8
RA	Risk Assessment	Management	5
SA	System and Services Acquistion	Management	14
SC	System and Communication Protection	Technical	34
SI	System and Information Integrity	Operational	13
PM	Program Management	Management	11

NIST also published SP 800-82, Guide to Industrial Control Systems (ICS) Security. In this publication they chose to provide supplemental guidance for several of the SP 800-53 controls. Much of this guidance provides compensatory controls for those controls that are difficult if not impossible to implement on an ICS.[13]

Vulnerabilities like the ones in mkdir, xterm, and fingerd do not have a clear home in this taxonomy. The closest place would Management/SA-8 Security Engineering Principles. This control simply states that security engineering principles are applied when software is developed or modified.

Vulnerabilities like the ones exposed in the Metcalf incident are covered in Operational/Contingency Planning and Operational/Physical Environment. The cable vulnerability would be addressed by physical environment controls that require a formal protection policy, access authorizations, access control, etc. The sniper vulnerability would by addressed by contingency planning controls which require that the organization have a plan, that employees are trained in that plan, and that the plan include recovery and reconstitution.

3.5 Engineering Principles

The NIST Special Publication 800-27 reviews 33 principles into 6 categories as illustrated in Table 4.[14]

Table 4 NIST principles

Categories	Principles
Security Foundations	4
Risk Based	7
Ease of Use	4
Increase Resilience	8
Reduce Vulnerabilities	6
Design with Network in Mind	4

Considering the 33 principles outlined in SP 800-27, we will review some of the vulnerabilites cited by Bishop to see if they fit uniquely. The mkdir vulnerability would fit into Reduce Vulnerabilities/Principle 29 “Identify and prevent common errors and vulnerabilities”.[14] One might want to bin the xterm vulnerability in Principle 29; however, it is most properly binned in Reduce Vulnerabilites/Principle 26 “Implement least privilege”.[14] The fingerd vulnerability could

be binned in Risk Based/Principle 6 “Assume the external systems are insecure.”[14] There is a great deal of overlap amongst the various principles and clearly this taxonomy would fail to be unique.

Looking at the vulnerabilies exploited in the Metcalf incident, we find the cable vulnerability could be addressed by Design with Network in Mind/Principle 30 “Implement security through a combination of measures distributed physically and logically.”[14] However promising the name may look, the text of the principle would not lead one to consider securing the entrance to the underground vault that contained the telephone wires. The sniper vulnerabilty could be addressed by Increase Resilience/Priniciple 23 “Develop and exercise contingency or disaster recovery prodecures to ensure appropriate availability.”[14] This principle does require plans to include the various phases of an event including “a return to normal operation phase.”[14] This taxonomy only provides a place for half of the vulnerabilities we selected from the Metcalf incident.

3.6 Protection Analysis Study

The Protection Analysis[2] study provides the following information:

1. Improper protection (initial and enforcement)
1a. improper choice of initial protection domain - “incorrect initial assignment of security or integrity level at system initialization or generation; a security critical function manipulating critical data directly accessible to the use”;
1b. improper isolation of implementation detail - allowing users to bypass operating system controls and write to absolute input/output addresses; direct manipulation of a “hidden” data structure such as a directory file being written to as if it were a regular file; drawing inferences from paging activity
1c. improper change - the “time-of-check to time-of-use” flaw; changing a parameter unexpectedly;
1d. improper naming - allowing two different objects to have the same name, resulting in confusion over which is referenced.
1e. improper deallocation or deletion - leaving old data in memory deallocated by one process and reallocated to another process, enabling the second process to access the information used by the first; failing to end a session properly
2. Improper validation - not checking critical conditions and parameters, leading to a process addressing memory not in its memory space by referencing through a out-of-bounds pointer value; allowing type clashes; overflows
3. Improper synchronization;
3a. improper indivisibility - interrupting atomic operations (e.g. locking); cache

inconsistency
3b. improper sequencing - allowing actions in an incorrect order (e.g. reading during writing)
4. Improper choice of operand or operation - using unfair scheduling algorithms that block certain processes or users from running; using the wrong function or wrong arguments.

Bishop has already demonstrated how to uniquely apply vulnerabilities to these classes;[2] however, there is no place in this taxonomy for the vulnerabilities exploited in the Metcalf incident.

3.7 Seven Pernicious Kingdoms

Tsipenyuk et al. divided the development vulnerability space into 7 plus 1 kingdoms and 85 phyla as shown in Table 5. They reviewed previous taxonomy attempts and show how these taxonomies map to theirs. They also demonstrate that their taxonomy is both more general and complete. Previous efforts focused on vulnerabilities in operating systems; whereas, this taxonomy expands to cover application coding errors including web application coding errors. This taxonomy is designed to incorporated into a static code analysis tool.[9]

Table 5 Seven pernicious kingdoms

Number	Kingdom	Phyla
1	Input Validation and Representation	26
2	API Abuse	11
3	Security Features	9
4	Time and State	7
5	Errors	4
6	Code Quality	9
7	Encapsulation	10
*	Environment	9

If we examine this taxonomy against Bishop's vulnerabilities, we find that each fits nicely into a kingdom and phylum. The mkdir vulnerability would be classified as a Time and State/File Access Race Condition. The xterm vulnerability would be classified as a Security Features/Least Privilege Violation. The fingerd vulnerability would be classified as Input Validation and Representation/Buffer Overflow. There is no place in this taxonomy for the vulnerabilities exploited in the Metcalf incident.

4. Proposal

The brief survey demonstrates that the vulnerability taxonomies seem to be too laser focused and are missing almost all vulnerabilities that are not directly related to software development, and the controls vulnerabilities are very broad and do not contain the level of detail required to capture software vulnerabilities in any reasonable manner. The obvious solution is to somehow combine one of the control taxonomies with one of the vulnerability taxonomies. Of the control taxonomies, the NSA IAM is too general and the DODI 8500.2 is obsolete leaving only the NIST 800-53, which is the most focused and most relevant of the control taxonomies. Of the vulnerability taxonomies, the "Seven Pernicious Kingdoms" seems to be the richest and most detailed.

Selecting the 2 taxonomies to merge only solves half of the problem. It is still necessary to find some way to combine them. The NIST 800-53 defined 3 kingdoms: management, operational, and technical. These kingdoms deal primarily with the operations of an information system and only briefly reference its development. What is necessary is another kingdom to encompass the development of the system. This would provide 4 kingdoms: management, operational, technical, and development. The 7 kingdoms would then become phyla under the development kingdom each containing several classes. The Seven Pernicious Kingdoms contain an eighth kingdom called environment, which is a bin for all of those vulnerabilities that are in the configuration of the system and not in the software development. Elements from the eighth kingdom will be assumed into one of the other kingdoms where they are already addressed.

In this taxonomy the 3 classes from NIST 800-53—management, technical, and operational—will be joined by a fourth class, development, to become the 4 kingdoms. The subject areas from the NIST 800-53 plus the kingdoms from the 7 kingdoms will all become phyla. The controls from NIST 800-53 and the phyla from the 7 kingdoms will become classes. At some point orders, families, genae, and species may need to be defined; however, if necessary that will addressed in future work.

This taxonomy may seem to have a problem with uniqueness. For example, the mkdir problem would have been place in one leaf of the tree before a patch or upgrade was available and in a different leaf of the tree a reasonable time after the patch or upgrade was available. These are too completely different vulnerabilities when one considers the big picture. The vulnerability that exists a reasonable amount of time after a patch or upgrade is available is a symptom of a failure

of the system that deploys these patches or upgrades. Failure to recognize this diverts the focus and resources from correcting the root causes to addressing the symptoms.

An outline of the proposed taxonomy is provided. Exact definitions and examples of the classes exist in the source documents and are not included here.

1. Management
    (a) CA Security Assessment and Authorization
        i. CA-1 Security Assessment and Authorization Policies and Procedures
        ii. CA-2 Security Assessments
        iii. CA-3 Information Systems Connections
        iv. CA-4 Security Certification (Withdrawn)
        v. CA-5 Plan of Action and Milestones
        vi. CA-6 Security Authorization
        vii. CA-7 Continuous Monitoring
    (b) PL Planning
        i. PL-1 Sercurity Planning Policy and Procedures
        ii. PL-2 System Security Plan
        iii. PL-3 System Security Plan Update (Withdrawn)
        iv. PL-4 Rules of Behavior
        v. PL-5 Privacy Impact Assessment
        vi. PL-6 Security-Related Activity Planning
    (c) PS Personnel Security
        i. PS-1 Personnel Security Policy and Procedures
        ii. PS-2 Position Categorization
        iii. PS-3 Personnel Screening
        iv. PS-4 Personnel Termination
        v. PS-5 Personnel Transfer
        vi. PS-6 Access Agreements
        vii. PS-7 Third-Pary Personnel Security
        viii. PS-8 Personnel Sanctions
    (d) RA Risk Assessment

i. RA-1 Risk Assessment Policy and Procedures
ii. RA-2 Security Categorization
iii. RA-3 Risk Assessment
iv. RA-4 Risk Assessment Update (Withdrawn)
v. RA-5 Vulnerability Scanning

(e) SA System and Services Acquisition

i. SA-1 System and Services Acquisition Policy and Procedures
ii. SA-2 Allocation of Resources
iii. SA-3 Life Cycle Support
iv. SA-4 Acquisitions
v. SA-5 Information System Documentation
vi. SA-6 Software Usage Restrictions
vii. SA-7 User-Installed Software
viii. SA-8 Security Engineering Principles
ix. SA-9 External Information System Services
x. SA-10 Developer Configuration Management
xi. SA-11 Developer Security Testing
xii. SA-12 Supply Chain Protection
xiii. SA-13 Trustworthiness
xiv. SA-14 Critical Information System Component

(f) PM Program Management

i. PM-1 Information Security Program Plan
ii. PM-2 Senior Information Security Officer
iii. PM-3 Information Security Resources
iv. PM-4 Plan of Action and Milestones Process
v. PM-5 Information System Inventory
vi. PM-6 Information Security Measures of Performance
vii. PM-7 Enterprise Architecture
viii. PM-8 Critical Infrastrure Plan
ix. PM-9 Risk Managment Strategy
x. PM-10 Security Authorization Process
xi. PM-11 Mission/Business Process Definition

2. Operational

   (a) AT Awareness and Training

      i. AT-1 Security Awareness and Training Policy and Procedures
      ii. AT-2 Security Awareness
      iii. AT-3 Security Training
      iv. AT-4 Security Training Records
      v. AT-5 Contacts with Security Groups and Associations

   (b) CM Configuration Management

      i. CM-1 Configuration Management Policy and Procedures
      ii. CM-2 Baseline Configuration
      iii. CM-3 Configuration Change Control
      iv. CM-4 Security Impact Analysis
      v. CM-5 Access Restrictions for Change
      vi. CM-6 Configuration Settings
      vii. CM-7 Least Functionality
      viii. CM-8 Information System Component Inventory
      ix. CM-9 Configuration Management Plan

   (c) CP Contingency Planning

      i. CP-1 Contingency Planning Policy and Procedures
      ii. CP-2 Contingency Plan
      iii. CP-3 Contingency Training
      iv. CP-4 Contingency Plan Testing and Exercise
      v. CP-5 Contingency Plan Update (Withdrawn)
      vi. CP-6 Alternate Storage Site
      vii. CP-7 Alternate Processing Site
      viii. CP-8 Telecommunications Services
      ix. CP-9 Information System Backup
      x. CP-10 Information System Recovery and Reconstitution

   (d) IR Incident Response

      i. IR-1 Incident Response Policy and Procedures
      ii. IR-2 Incident Response Training

iii. IR-3 Incident Response Testing and Exercise
iv. IR-4 Incident Handling
v. IR-5 Incident Monitoring
vi. IR-6 Incident Reporting
vii. IR-7 Incident Response Assistance
viii. IR-8 Incident Response Plan

(e) MA Maintenance

i. MA-1 System Maintenance Policy and Procedures
ii. MA-2 Controlled Maintenance
iii. MA-3 Maintenance Tools
iv. MA-4 Non-Local Maintenance
v. MA-5 Maintenance Personnel
vi. MA-6 Timely Maintenance

(f) MP Media Protection

i. MP-1 Media Protection Policy and Procedures
ii. MP-2 Media Access
iii. MP-3 Media Marking
iv. MP-4 Media Storage
v. MP-5 Media Transport
vi. MP-6 Media Sanitization

(g) PE Physical Environment Protection

i. PE-1 Physical and Environmental Protection Policy and Procedures
ii. PE-2 Physical Access Authorization
iii. PE-3 Physical Access Control
iv. PE-4 Access Control for Transmission Medium
v. PE-5 Access Control for Output Devices
vi. PE-6 Monitoring Physical Access
vii. PE-7 Visitor Control
viii. PE-8 Access Records
ix. PE-9 Power Equipment and Power Cabling
x. PE-10 Emergency Shutoff

xi. PE-11 Emergency Power
xii. PE-12 Emergency Lighting
xiii. PE-13 Fire Protection
xiv. PE-14 Temperature and Humidity Controls
xv. PE-15 Water Damage Protection
xvi. PE-16 Delivery and Removal
xvii. PE-17 Alternate Work Site
xviii. PE-18 Location of Information System Components
xix. PD-19 Information Leakage

(h) SI System and Information Integrity

i. SI-1 System and Information Integrity Policy and Procedures
ii. SI-2 Flaw Remediation
iii. SI-3 Malicious Code Protection
iv. SI-4 Information System Monitoring
v. SI-5 Security Alerts, Advisories, and Directives
vi. SI-6 Security Functionality Verification
vii. SI-7 Software and Information Integrity
viii. SI-8 Spam Protection
ix. SI-9 Information Input Restrictions
x. SI-10 Information Input Validation
xi. SI-11 Error Handling
xii. SI-12 Information Ouput Handling and Retention
xiii. SI-13 Predictable Failure Prevention

3. Technical

(a) AC Access Control

i. AC-1 Acess Control Policy and Procedures
ii. AC-2 Account Management
iii. AC-3 Access Enforcement
iv. AC-4 Information Flow Enforcement
v. AC-5 Separation of Duties
vi. AC-6 Least Privilege

vii. AC-7 Unsuccessful Login Attempts
viii. AC-8 System Use Notification
ix. AC-9 Previous Login (Access) Notification
x. AC-10 Concurrent Session Control
xi. AC-11 Session Lock
xii. AC-12 Session Termination (Withdrawn)
xiii. AC-13 Supervision and Review—Access Control (Withdrawn)
xiv. AC-14 Permitted Actions without Identification or Authentication
xv. AC-15 Automated Marking (Withdrawn)
xvi. AC-16 Security Attributes
xvii. AC-17 Remote Access
xviii. AC-18 Wireless Access
xix. AC-19 Access Control for Mobile Devices
xx. AC-20 Use of External Information Systems
xxi. AC-21 User-Based Collaboration and Information Sharing
xxii. AC-22 Publicly Accessible Content

(b) AU Audit and Accountability

i. AU-1 Audit and Accountability Policy and Procedures
ii. AU-2 Auditable Events
iii. AU-3 Content of Audit Records
iv. AU-4 Audit Storage Capacity
v. AU-5 Response to Audit Processing Failures
vi. AU-6 Audit Review, Analysis, and Reporting
vii. AU-7 Audit Reduction and Report Generation
viii. AU-8 Time Stamps
ix. AU-9 Protection of Audit Information
x. AU-10 Non-repudiation
xi. AU-11 Audit Record Retention
xii. AU-12 Audit Generation
xiii. AU-13 Monitoring for Information Disclosure
xiv. AU-14 Session Audit

(c) IA Identification and Authentication

i. IA-1 Identification and Authentication Policy and Procedures
ii. IA-2 Identification and Authentication (Organizational Users)
iii. IA-3 Device Identification and Authentication
iv. IA-4 Identifier Management
v. IA-5 Authenticator Management
vi. IA-6 Authenticator Feedback
vii. IA-7 Cryptographic Module Authentication
viii. IA-8 Identification and Authentication (Non-Organizational Users)

(d) SC System and Communication Protection

i. SC-1 System and Communications Protection Policy and Procedures
ii. SC-2 Application Partitioning
iii. SC-3 Security Function Isolation
iv. SC-4 Information in Shared Resources
v. SC-5 Denial of Service Protection
vi. SC-6 Resource Priority
vii. SC-7 Boundary Protection
viii. SC-8 Transmission Integrity
ix. SC-9 Transmission Confidentiality
x. SC-10 Network Disconnect
xi. SC-11 Trusted Path
xii. SC-12 Cryptographic Key Establishment and Management
xiii. SC-13 Use of Cryptography
xiv. SC-14 Public Access Protections
xv. SC-15 Collaborative Computing Devices
xvi. SC-16 Transmission of Security Attributes
xvii. SC-17 Public Key Infrastructure Certificates
xviii. SC-18 Mobile Code
xix. SC-19 Voice Over Internet Protocol
xx. SC-20 Secure Name/Address Resolution Service (Authoritative Source)
xxi. SC-21 Secure Name/Address Resolution Service (Recursive or Caching Resolver)
xxii. SC-22 Architecture and Provisioning for Name/Address Resolution Service
xxiii. SC-23 Session Authenticity

xxiv. SC-24 Fail in Known State
xxv. SC-25 Thin Nodes
xxvi. SC-26 Honeypots
xxvii. SC-27 Operating System-Independent Applications
xxviii. SC-28 Protection of Information at Rest
xxix. SC-29 Heterogeneity
xxx. SC-30 Virtualization Techniques
xxxi. SC-31 Covert Channel Analysis
xxxii. SC-32 Information System Partitioning
xxxiii. SC-33 Transmission Preparation Integrity
xxxiv. SC-34 Non-Modifiable Executable Programs

4. Development

(a) Input Validation and Representation

i. Buffer Overflow
ii. Command Injection
iii. Cross-Site Scripting
iv. Format String
v. HTTP Response Splitting
vi. Illegal Pointer Value
vii. Integer Overflow
viii. Log Forging
ix. Path Manipulation
x. Process Control
xi. Resource Injection
xii. Setting Manipulation
xiii. SQL Injection
xiv. String Termination Error
xv. Struts: Duplicate Validation Forms
xvi. Struts: Erroneous validate() Method
xvii. Struts: Form Bean Does Not Extend Validation Class
xviii. Struts: Form Field Without Validator

xix. Struts: Plug-in Framework Not in Use
xx. Struts: Unused Validation Form
xxi. Struts: Unvalidated Action Form
xxii. Struts: Validator Turned Off
xxiii. Struts: Validator Without From Field
xxiv. Unsafe JNI
xxv. Unsafe Reflection
xxvi. XML Validation

(b) API Abuse

i. Dangerous Function
ii. Directory Restriction
iii. Heap Inspection
iv. J2EE Bad Practices: getConnection().
v. J2EE Bad Practices: Sockets.
vi. Often Misused: Authentication
vii. Often Misused: Exception Handling
viii. Often Misused: File System
ix. Often Misused: Privilege Management
x. Often Misused: Strings
xi. Unchecked Return Value

(c) Security Features

i. Insecure Randomness
ii. Least Privilege Violation
iii. Missing Access Control
iv. Password Management
v. Password Management: Empty Password in Config File
vi. Password Management: Hard-Coded Password
vii. Password Management: Password in Config File
viii. Password Management: Weak Cryptography
ix. Privacy Violation

(d) Time and State

i. Deadlock

ii. Failure to Begin a New Session upon Authentication
iii. File Access Race Condition: TOCTOU
iv. Insecure Temporary File
v. J2EE Bad Practice: System.exit()
vi. J2EE Bad Practice: Threads
vii. Signal Handling Race Conditions

(e) Errors

i. Catch
ii. Empty Catch Block
iii. Overly-Broad Catch Block
iv. Overly-Broad Throws Declaration

(f) Code Quality

i. Double Free
ii. Inconsistent Implementations
iii. Memory Leak
iv. Null Dereference
v. Obsolete
vi. Undefined Behavior
vii. Uninitialized Variable
viii. Unrelease Resource
ix. Use After Free

(g) Encapsulation

i. Comparing Classes by Name
ii. Data Leaking between Users
iii. Leftover Debug Code
iv. Mobile Code: Object Hijack
v. Mobile Code: Use of Inner Class
vi. Mobile Code: Non-Final Public Field
vii. Private Array-Typed Field Returned from a Public Method
viii. Public Data Assigned to Private Array-Typed Field
ix. System Information Leak
x. Trust Boundary Violation

5. Conclusions and Future Work

Most of the taxonomies for vulnerabilities suffer myopia in that they are too strictly focused on one area of concern to be of general use. Taxonomies of controls are broad enough to be of general use; however, they are focused on the postdeployment of the system and have little support for the development of the systems. In many cases there does not appear to be a control for a vulnerability until after there is a fix; either an upgrade or a malicious code signature. Adding an existing very specific vulnerability taxonomy to a control taxonomy may provide the breadth and depth necessary in a general taxonomy. Although the genesis of this work was to develop a taxonomy for SCADA systems, SCADA systems are a specialized form of IT, and the community would be better served by adding any SCADA specific vulnerabilities to a broader IT taxonomy than to develop yet another myopic taxonomy.

The next step is to take this taxonomy and analyze each vulnerability or control for existing counter measures for the purpose of determining the residual risk. This will enable us to discover unacceptable gaps. These gaps may be used to create research direction.

6. References

1. History of taxonomy. http://www.atbi.eu/summerschool/files/summerschool/Manktelow_Syllabus.pdf [accessed 2014 Aug 26].

2. Bishop M. A taxonomy of unix system and network vulnerabilities. Department of Computer Science, University of California at Davis; 1995 May . Report No.: CSE-95-10.

3. Igure VM. A taxonomy of security vulnerabilities in SCADA protocols [thesis]. University of Virginia, 2007.

4. Bishop M, Bailey D. A critical analysis of vulnerability taxonomies. Defense Technical Information Center; September 1996. Document No.: CSE-96-11.

5. Smith R. Assault on california power station raises alarm on potential for terrorism. http://online.wsj.com/news/articles/SB10001424052702304851104579359141941621778 [accessed 2014 Aug 26].

6. Igure V, Williams R. Taxonomies of attacks and vulnerabilities in computer systems. Communications Surveys & Tutorials, IEEE. 2008;10(1):6–19.

7. CVE List Main Page. https://cve.mitre.org/cve [accessed October 2014].

8. Zhu B, Joseph A, Sastry S. A taxonomy of cyber attacks on SCADA systems. In: Internet of Things (iThings/CPSCom), 2011 International Conference on Cyber, Physical and Social Computing; 2011 October 19–22; Dalian, China. IEEE; 2011. p. 380–388.

9. Tsipenyuk K, Chess B, McGraw G. Seven pernicious kingdoms: A taxonomy of software security errors. Security & Privacy, IEEE. 2005;3(6):81–84.

10. Rogers R, Miles G, Dykstra T, Fuller E. Security assessment: case studies for implementing the NSA IAM. Rockland (MA): Syngress Publishing; 2004.

11. Ross R, Katzke S, Johnson A, Swanson M, Stoneburner G, Rogers G, Lee A. Recommended security controls for federal information systems. Gaithersburg (MD): National Institute of Standards and Technology; 2005. Special Publication No.: 800-53.

12. Department of defense information assurance (IA) implementation. Washington (DC): Department of Defense; 2003 Feb. Instruction No.: 8500.2.

13. Stouffer K, Falco J, Scarfone K. Guide to industrial control systems (ICS) security. Gaithersburg (MD): National Instititute of Standards and Technology; 2011. Special Publication No.: 800-82.

14. Stoneburner G, Hayden C, Feringa A. Engineering principles for information technology security (a baseline for achieving security). Gaithersburg (MD): National Institute of Standards and Technology; 2001. Special Publication No.: 800-27.

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIRECTOR US ARMY RESEARCH LAB RDRL CIO LL IMAL HRA MAIL & RECORDS MGMT
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
1 (PDF)	DIR USARL RDRL CIN S S SMITH

INTENTIONALLY LEFT BLANK.